Hochschule Fulda
University of Applied Sciences

Exercise 3 - Parallel Programming

Course: Parallel Programming, Wintersemester 2024/25
Anna-Lena Roth

Date: 01.11.2025

1) In the first exercise, you have developed a program that calculates an approximation to the value Pi (π) using the Leibniz formula. You now want to parallelize it using OpenMP.
    a. Add parallel region and for directive and run the code with OMP_NUM_THREADS=1. It is expected that Pi will be calculated correctly.
    b. Run the code again with OMP_NUM_THREADS=4. It is expected that Pi will not be calculated correctly. Repeat also with different OMP_NUM_THREADS values. Why is it unpredictable? (Where is a race condition)?
    c. Add a critical directive around sum statement. Compile and run. Test different different OMP_NUM_THREADS several times in a row.
        i. How is the speedup with increasing OMP_NUM_THREADS? (why do e.g. 4 threads take longer than 2?)
        ii. Compare results. Are the results the same to the last digit? Why not?
    d. Optimize: Move critical region outside loop.
    Tip: Each thread should calculate its own partial sum in parallel. Finally, the partial sums of the threads within a critical region are then added. Now check changes to the speedup again.